Requirement Metrics for Risk Identification

Theodore Hammer, GSFC, 301-614-5225, Theodore.Hammer@gsfc.nasa.gov
Lenore Huffman, SATC, 301-286-0099, Lenore.Huffman@gsfc.nasa.gov
William Wilson, SATC, 301-286-0102, William.Wilson@gsfc.nasa.gov
Dr. Linda Rosenberg, SATC, 301-286-0087, Linda.Rosenberg@gsfc.nasa.gov
Lawrence Hyatt, GSFC, 301-286-7475, Larry.Hyatt@gsfc.nasa.gov

1. Introduction

The Software Assurance Technology Center (SATC) is part of the Office of Mission Assurance of the Goddard Space Flight Center (GSFC). The SATC's mission is to assist National Aeronautics and Space Administration (NASA) projects to improve the quality of software which they acquire or develop. The SATC's efforts are currently focused on the development and use of metric methodologies and tools that identify and assess risks associated with software performance and scheduled delivery. This starts at the requirements phase, where the SATC, in conjunction with software projects at GSFC and other NASA centers is working to identify tools and metric methodologies to assist project managers in identifying and mitigating risks. This paper discusses requirement metrics currently being used at NASA in a collaborative effort between the SATC and the Quality Assurance Office at GSFC to utilize the information available through the application of requirements management tools.

Requirements development and management have always been critical in the implementation of software systems - engineers are unable to build what analysts can not define. Recently, automated tools have become available to support requirements management. The use of these tools not only provides support in the definition and tracing of requirements, but also opens the door to effective use of metrics in characterizing and assessing risks. Metrics are important because of the benefits associated with early detection and correction of problems with requirements; problems not found until testing are at least 14 times more costly to fix than problems found in the requirements phase. This paper discusses two facets of the SATC's efforts to identify requirement risks early in the life cycle, thus preventing costly errors and time delays later in the life cycle.

The first effort that will be discussed is the development and application of an early life cycle tool for assessing requirements that are specified in natural language. This paper describes the development and experimental use of the Automated Requirements Measurement (ARM) tool. Reports produced by the tool are used to identify specification statements and structural areas of the requirements document which need to be improved.

The second effort discusses metrics analysis of information in the requirements database used to provide insight into the stability and expansion of requirements. The research into attaching certain document attributes to analyses results done on requirements stored in requirements databases is providing project management with valuable information. The correlations between document structure and language, and requirement expansion and

testing have been strong. This information has been assisting and continues to assist the Quality Assurance Office in its project oversight role.

When discussing metric results the project must remain anonymous; however, for this paper, a general understanding of the project's development environment is necessary. The project in discussion is implementing a large system in three main incremental builds.[1] The development of these builds is overlapping, e.g. design and coding of the second and third builds started prior to the completion of the first build. Each build adds new functionality to the previous build and satisfies a further set of requirements. The definition of requirements for this system started with the formulation of System Level Requirements. These are mission level requirements for the space craft and ground system; they are at a very high level and rarely, if ever, change. Requirements at this level will not be discussed since they are not stored in the requirements database under scrutiny.

System requirements then undergo several levels of decomposition to produce Top Level Requirements. These requirements are also high level and change should be minimal. The development of the project discussed in this paper started with the Top Level requirements. Top Level requirements are then divided into subsystems and a further level is derived in greater detail; hence, "Specification Requirements". Generally, contracts are bid using this level of requirement detail. The Design Requirements are derived from the Specification requirements; these requirements are the ones used to design and code the system. This project chose to develop an additional intermediate set of Specification Level Requirements after contract award.

2. Automated Requirements Measurement Tool (ARM)

Despite the significant advantages attributed to the use of formal specification languages, their use has not become common practice. Because requirements that the acquirer expects the developer to contractually satisfy must be understood by both parties, specifications are most often written in natural language. The use of natural language to prescribe complex, dynamic systems has at least three severe problems: ambiguity, inaccuracy and inconsistency. Many words and phrases have dual meanings which can be altered by the context in which they are used. Weak sentence structure can also produce ambiguous statements. For example, the statement "Twenty seconds prior to engine shutdown anomalies shall be ignored." could result in at least three different implementations. Defining a large, multi-dimensional capability within the limitations imposed by the two dimensional structure of a document can obscure the relationships between individual groups of requirements.

The SATC developed the Automated Requirements Measurement (ARM) tool to address certain management needs: that of providing metrics which NASA project managers can use to assess the quality of their requirements specification documents and that of identifying risks poorly specified requirements introduce into any project. The ARM tool searches the requirements document for terms the SATC has identified as quality indicators. Reports produced by the tool are used to identify specification statements and structural areas of the

[1] Various names are used, deliveries, releases, builds, but the term build will be used in this paper.

requirements document which need improvement. It must be emphasized that the tool does not assess correctness of the requirements specified; it does, however, assess the structure, language, and vocabulary of both the document itself and the individual requirements.

2.1 Specification Quality Attributes

The SATC study was initiated by compiling the following list of quality attributes that requirements specifications are expected to exhibit: Completeness, Consistency, Correctness, Modifiability, Ranking, Traceability, Non-ambiguity, and Verifiability. As a practical matter, it is generally accepted that requirements specifications should also be Valid and Testable. These characteristics are not independent. A specification, obviously, cannot be correct if it is incomplete or inconsistent.

Most, if not all, of these quality attributes are subjective. A conclusive assessment of a requirements specification's appropriateness requires review and analysis by technical and operational experts in the domain addressed by the requirements. Several of these quality attributes, however, can be linked to primitive indicators that provide some evidence that the desired attributes are present or absent.

2.2 Specification Quality Indicators

Although most of the quality attributes of documented requirements are subjective, there are aspects of the documentation which can be measured and therefore can be used as indicators of quality attributes. Nine categories of quality indicators for requirement documents and specification statements were established for two types of classification: those related to the examination of individual specification statements, and those related to the requirements document as a whole. The categories related to individual specification statements are: Imperatives, Continuances, Weak Phrases, Directives, and Options. The categories of indicators related to the entire requirements document are: Size, Specification Depth, Readability, and Text Structure.

- IMPERATIVES are those words and phrases that command that something must be provided. "Shall" normally dictates the provision of a functional capability; "Must" or "must not" normally establishes performance requirements or constraints; "Will" normally indicates that something will be provided from outside the capability being specified. The ARM report lists the imperatives and their associated counts in descending order of forcefulness. An explicit specification will have most of its counts high in the report IMPERATIVE list (i.e. shall, must, required).
- CONTINUANCES are phrases such as "the following:" that follow an imperative and precede the definition of lower level requirement specification. The extent that CONTINUANCES are used is an indication that requirements have been organized and structured. These characteristics contribute to the tractability and maintenance of the subject requirement specification. However, extensive use of continuances indicate multiple, complex requirements that may not be adequately factored into development resource and schedule estimates.

- WEAK PHRASES are clauses that are apt to cause uncertainty and leave room for multiple interpretations. Use of phrases such as “adequate” and “as appropriate” indicate that what is required is either defined elsewhere or worst, the requirement is open to subjective interpretation. Phrases such as “but not limited to” and “as a minimum” provide the basis for expanding requirements that have been identified or adding future requirements. WEAK PHRASE total is indication of the extent that the specification is ambiguous and incomplete.
- DIRECTIVES are words or phrases that indicate that the document contains examples or other illustrative information. DIRECTIVES point to information that makes the specified requirements more understandable. The implication is the higher the number of Total DIRECTIVES the more precisely the requirements are defined.
- OPTIONS are those words that give the developer latitude in the implementation of the specification that contains them. This type of statement loosens the specification, reduces the acquirer’s control over the final product, and establishes a basis for possible cost and schedule risks.
- LINES OF TEXT are the number of individual lines of text read by the ARM program from the source file.
- UNIQUE SUBJECTS is the count of unique combinations and permutations of words immediately preceding imperatives in the source file. This count is an indication of the scope of the document. The ratio of unique subjects to the total for SPECIFICATION STRUCTURE is also an indicator of the specifications’ detail.
- READABILITY STATISTICS are a category of indicators that measure how easily an adult can read and understand the requirements document. Flesch-Kincaid Grade Level index is also based on the average number of syllables per word and the average number of words per sentence. (For the project of this paper, the score indicates a grade school level.)

Table 1 below shows the summary statistics for 41 NASA requirement documents and the results for the project discussed in this paper, Project X.

41 DOCUMENTS	**Lines of Text** - Count of the physical lines of text	**Imperatives** - shall, must, will, should, is required to, are applicable, responsible for	**Continuances** - as follows, following, listed, inparticular, support	**Weak Phrases** - adequate, as applicable, as appropriate, as a minimum, be able to, be capable, easy, effective, not limited to, if practical	**Directives** - figure, table, for example, note:	**Options** - can, may, optionally	**Subjects** - Count fo unique identifiers preceding imperatives	**Flesch-Kincaid Grade Lvl** - Readability Index
Median	**927.0**	**192.5**	**70.5**	**24.0**	**13.0**	**20.0**	**76.0**	**11.4**
Mean	**1,569.9**	**415.1**	**155.4**	**41.0**	**25.0**	**39.6**	**173.5**	**10.8**
Max	**7,499.0**	**2,004.0**	**1,023.0**	**249.0**	**119.0**	**169.0**	**804.0**	**13.8**
Min	**36.0**	**25.0**	**8.0**	**0.0**	**0.0**	**0.0**	**12.0**	**7.8**
Stdev	**1,758.3**	**468.7**	**202.2**	**51.3**	**28.7**	**45.4**	**203.2**	**1.6**
Project X	**11,596**	**1,982**	**620**	**374**	**132**	**177**	**510**	**9**

Table 1: Summary Statistics

Two approaches can be applied to compare Project X to the metric database containing 41 documents. The first approach is to compare Project X to the other projects using standard deviations. Since approximately 99% of the projects should fall within +/- 3 standard deviations, we mark that range on the graph in Figure 1.

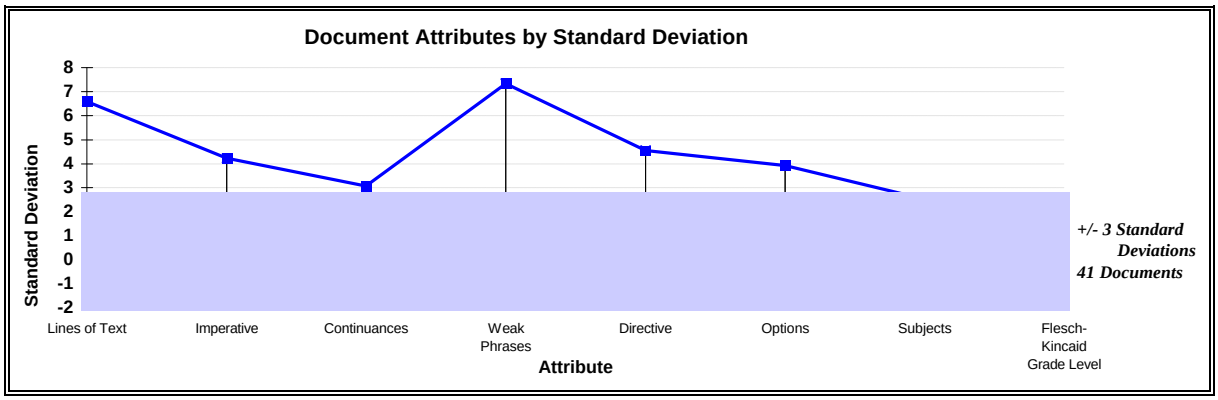

Figure 1: Document Attributes by Standard Deviation

However, since Project X is larger than all projects analyzed to date, Figure 1 may present an inaccurate picture. Normalizing the data on Lines of Text (Figure 2) yields a different picture of Project X in relation to other projects, thus suggesting that Project X attribute counts are in line. The number of weak phrases should however be investigated since it indicates potential risk.

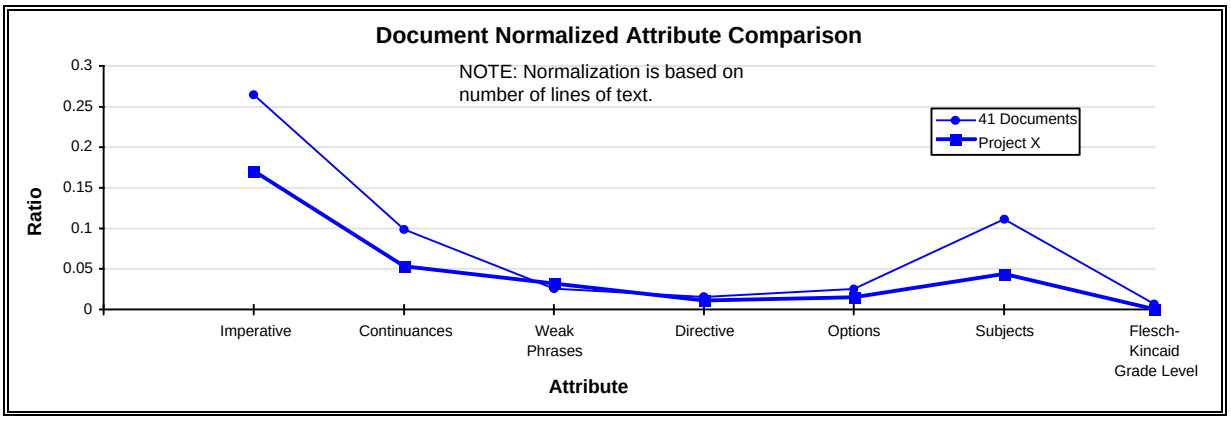

Figure 2: Document Attributes Normalized by Lines of Text

The structure of the document is also indicative of potential project risks. ARM uses the structure depth and specification depth to depict two aspects of the document's structure.

- STRUCTURE DEPTH provides a count of the numbered statements at each level of the source document. These counts provide an indication of the document's organization and consistency and level of detail. High level specifications will usually not have numbered

statements below a structural depth of four. Detailed documents may have numbered statements down to a depth of nine. A document that is well organized and maintains a consistent level of detail will have a pyramidal shape (few numbered statements at level 1 and each lower level having more numbered statements than the level above it). Documents that have an hour-glass shape (many numbered statements at high levels, few at mid levels and many at lower levels) are usually those that contain a large amount of introductory and administrative information. Diamond shaped documents (a pyramid followed by decreasing statement counts at levels below the pyramid) indicate that subjects introduced at the higher levels are probably addressed at different levels of detail.

- SPECIFICATION DEPTH is a count of the number of imperatives at each level of the document. These numbers also include the count of lower level list items that are introduced at a higher level by an imperative that is followed by a continuance. This structure has the same implications as the numbering structure. However, it is significant because it reflects the structure of the requirements as opposed to that of the document. Differences between the shape of the numbering and specification structure are an indication of the amount and location of background and/or introductory information is included in the document. The ratio of total for SPECIFICATION STRUCTURE to total lines of text is an indication of how concise the document is in specifying requirements.

The application of this information is still under investigation, and initial results from Project X are interesting. Figure 3 depicts expected structure versus actual structure of the Specification and Design requirement documents. The project data suggests the Specification requirements may have been overly defined, therefore artificially constraining the design and its expansion. The structure of the imperative levels in the Design document reinforces this observation, indicating little expansion where extensive expansion is expected.

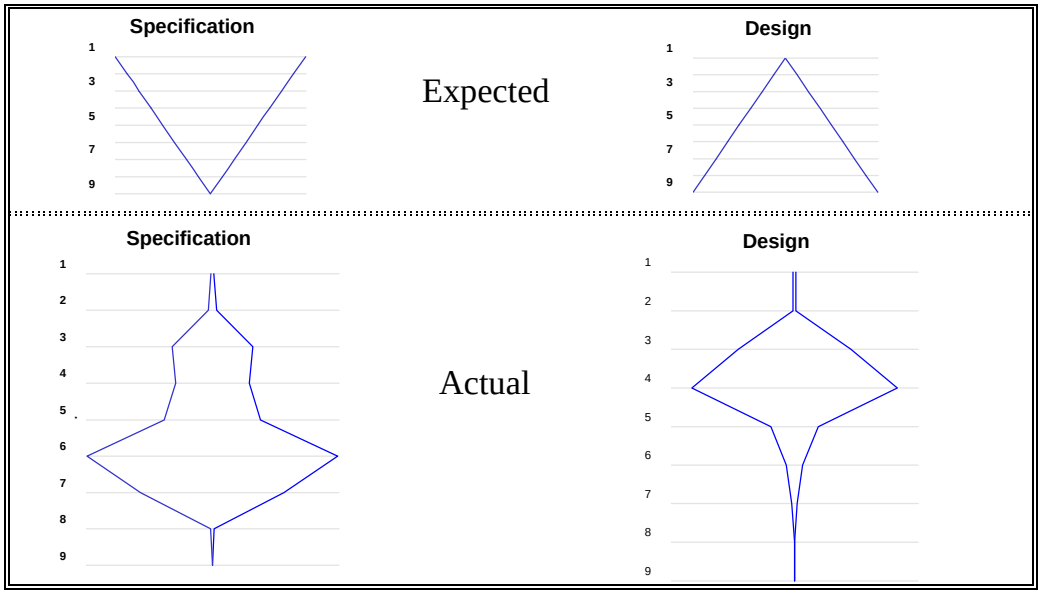

Figure 3: Document Depth at Which Imperatives are Located

3. Requirement Metrics

This section of the paper focuses on the application of metrics available through the use of a requirements management CASE tool. These metrics assist project managers and quality assurance engineers to identify the risks of insuring that the completed software system contains the functionality specified by the requirements. There are no published or industry standard guidelines for these metrics: intuitive interpretations, based on experience and supported by project feedback, are used in this paper. Project management has reacted favorably to the metrics and has used the analysis results to mitigate certain perceived risks. The SATC continues working on methods to mathematically validate the intuitive guidelines so that the requirement metrics and their interpretation are applicable to an ever increasing variety of software development applications. Three areas of requirement metrics will be discussed: Stability Over Time Per Requirement Design Level, Stability Over Time By Project Build, Expansion From Specification To Design Level.

3.1 Requirement Stability Over Time per Requirement Design Level

Requirements are developed and baselined at major reviews during the system development life cycle. At these milestone reviews, documents containing the requirements are reviewed and commented upon. After resolution of the comments, the requirement documents are baselined and put under configuration control. Ideally, the rate of change in each level of requirements should decrease as a milestone review approaches. Figure 4 shows the count of requirements at each level during the 6 month period starting at Preliminary Design Level (PDR) (through Critical Design Review (CDR) As expected, the Top Level and Specification requirements remained stable during this six month period. The Intermediate Specification and Design Level documents both stabilized prior to CDR.

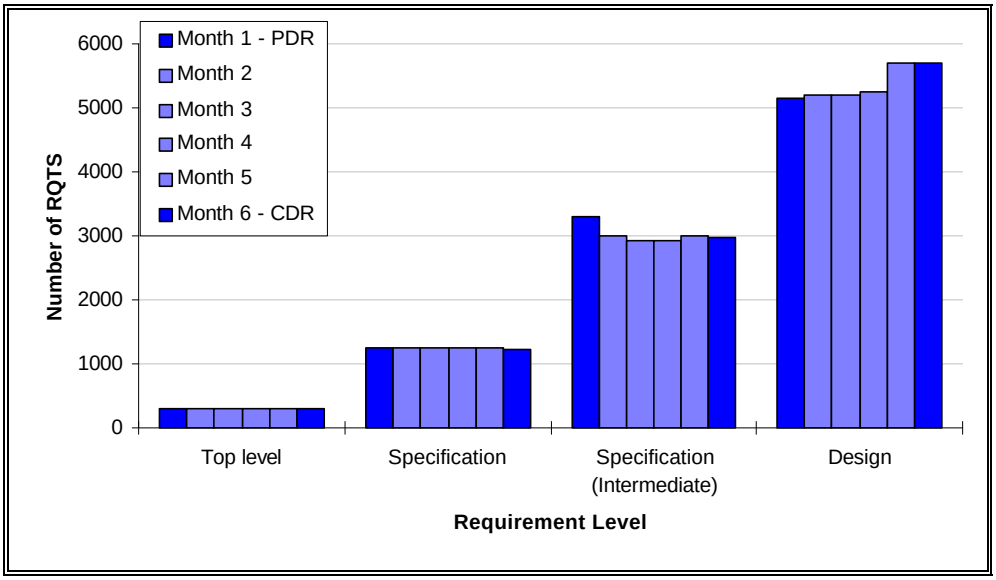

Figure 4: Requirement Count by Document Level

3.2 Requirement Stability Over Time By Project Build

As stated earlier, this system is implemented in three builds with the Specification and Design requirements allocated to each of these builds. One of the purposes of a multiple build development effort is to minimize the implementation risk associated with any one build. This insures that no single build implements an inordinate number of requirements. Figure 5 shows the counts of the Design Requirements (Figure 4) for Build 1 and Build 2.

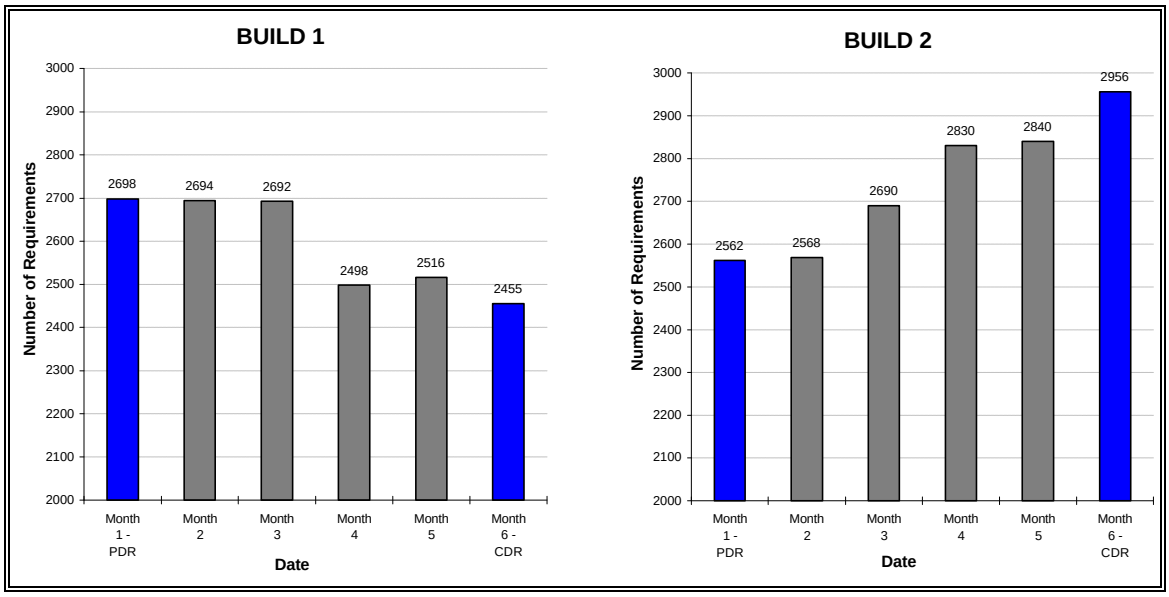

Figure 5: Design Requirement Allocation by Build

This is a different picture of the requirements stability, showing a shift in the number of requirements from Build 1 to Build 2; and indicating a potential risk to the schedule of Build 2 which should be closely monitored.

3.3 Requirement Detail Expansion

In addition to requirement stability, the expansion of the upper level requirements to more detailed levels generates potential project risk. Figure 6 shows the number of requirements of the Detail level referencing the number of requirements in the Intermediate Specification Level. The tails on the expected curve in the upper right indicate a scattering of upper level requirements referenced either by very few or very many detailed requirements; however, the majority of requirements will have multiple references and result in a bell-shaped curve..

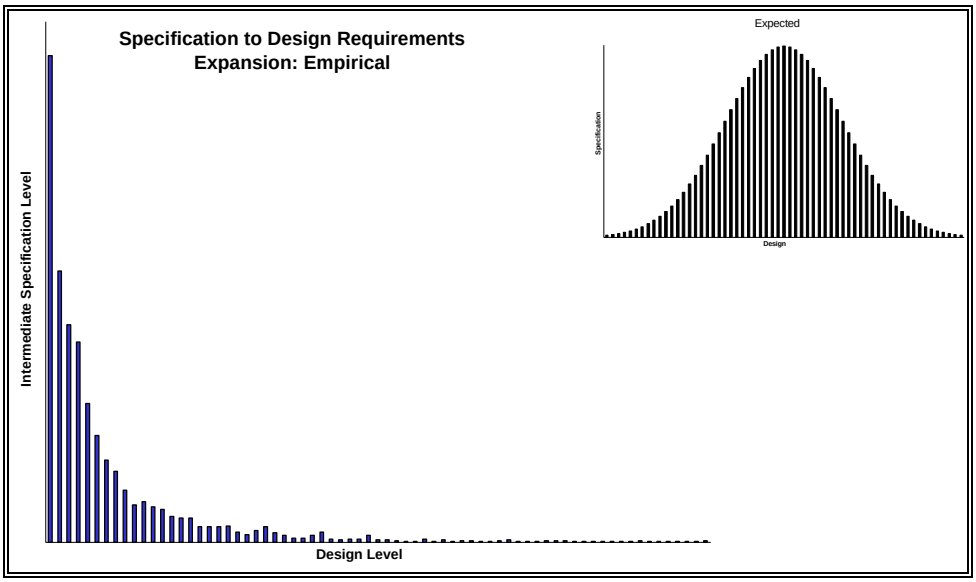

Figure 6: Requirement Expansion

Project data however, does not match the expected; there is a high number of Intermediate Specification requirements that are referenced by only one or very few Detail requirements (left hand side of the graph) while other requirements have high numbers of multiple references (right hand side of graph). As an example, the Intermediate Specification requirement "The system shall have a database." is linked to 200 Design level requirements. The shape of the curve in Figure 6 indicates that the Design analysis is incomplete with the specific requirements are not adequately decomposed, thus suggesting that requirements were copied with neither analysis nor expansion into detail for the implementation phase.

4. Conclusions

Based on the work done to date, four conclusions can be reached:

- Requirement metrics assist in identifying potential project risks
- Multiple metrics are needed for comprehensive evaluation
- Evaluation of requirement text can yield risk information very early in the life cycle
- Metric collection is cheaper, faster and more reliable with requirement management tools

Using automated tools to track requirements has opened the door to deriving metrics for characterizing requirement text , stability and expansion rate. Tracking and correlating test cases and test results to individual requirements within a database is essential for viewing relationships not otherwise available.. The use of an automated requirements database allows the metrics program to generate metrics for best insight into the requirements and test case interplay . The metrics presented in this paper are the result of much research into data use and pictorial display; however, all results have been used by project management to successfully identify and manage risks.

REFERENCES

Brooks, Frederick P. Jr., No Silver Bullet: Essence and accidents of software engineering, *IEEE Computer, vol. 15, no. 1*, April 1987, pp. 10-18.

DOD MIL-STD-490A, Specification Practices, June 4, 1985.

IEEE Std 830-1993, Recommended Practice for Software Requirements Specifications, December 2, 1993.

Kitchenham, Barbara, Pfleeger, Shari Lawrence, Software Quality: The Elusive Target, *IEEE Software, Vol. 13, No. 1*, January 1996, pp. 12-21.`

Stokes, David Alan, Requirements Analysis, *Computer Weekly Software Engineer's Reference Book*, 1991, pp. 16/3-16/21.

Wilson, William, Rosenberg, Linda, Hyatt, Lawrence, Automated Quality analysis of Natural Language Requirement Specifications, Fourteen Annual Pacific Northwest Software Quality Conference, October, 1996.